R Notebook

This is an [R Markdown](http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Cmd+Shift+Enter*.

plot(cars)



Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Cmd+Option+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Cmd+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.5.1 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.4 ✔ tidyr 1.3.1  
## ✔ purrr 1.0.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

# loading the tidyverse package  
data()   
# displays the list of data sets available in R  
glimpse(msleep)

## Rows: 83  
## Columns: 11  
## $ name <chr> "Cheetah", "Owl monkey", "Mountain beaver", "Greater shor…  
## $ genus <chr> "Acinonyx", "Aotus", "Aplodontia", "Blarina", "Bos", "Bra…  
## $ vore <chr> "carni", "omni", "herbi", "omni", "herbi", "herbi", "carn…  
## $ order <chr> "Carnivora", "Primates", "Rodentia", "Soricomorpha", "Art…  
## $ conservation <chr> "lc", NA, "nt", "lc", "domesticated", NA, "vu", NA, "dome…  
## $ sleep\_total <dbl> 12.1, 17.0, 14.4, 14.9, 4.0, 14.4, 8.7, 7.0, 10.1, 3.0, 5…  
## $ sleep\_rem <dbl> NA, 1.8, 2.4, 2.3, 0.7, 2.2, 1.4, NA, 2.9, NA, 0.6, 0.8, …  
## $ sleep\_cycle <dbl> NA, NA, NA, 0.1333333, 0.6666667, 0.7666667, 0.3833333, N…  
## $ awake <dbl> 11.9, 7.0, 9.6, 9.1, 20.0, 9.6, 15.3, 17.0, 13.9, 21.0, 1…  
## $ brainwt <dbl> NA, 0.01550, NA, 0.00029, 0.42300, NA, NA, NA, 0.07000, 0…  
## $ bodywt <dbl> 50.000, 0.480, 1.350, 0.019, 600.000, 3.850, 20.490, 0.04…

# provides a quick overview of the data set  
head(msleep)

## # A tibble: 6 × 11  
## name genus vore order conservation sleep\_total sleep\_rem sleep\_cycle awake  
## <chr> <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 Cheetah Acin… carni Carn… lc 12.1 NA NA 11.9  
## 2 Owl mo… Aotus omni Prim… <NA> 17 1.8 NA 7   
## 3 Mounta… Aplo… herbi Rode… nt 14.4 2.4 NA 9.6  
## 4 Greate… Blar… omni Sori… lc 14.9 2.3 0.133 9.1  
## 5 Cow Bos herbi Arti… domesticated 4 0.7 0.667 20   
## 6 Three-… Brad… herbi Pilo… <NA> 14.4 2.2 0.767 9.6  
## # ℹ 2 more variables: brainwt <dbl>, bodywt <dbl>

# prints the first 6 rows of data set

# with respective variable(awake)  
min(msleep$awake)

## [1] 4.1

# gets the minimum of awake variable  
max(msleep$awake)

## [1] 22.1

# gets the maximum of awake variable  
range(msleep$awake)

## [1] 4.1 22.1

# gets the range of values  
IQR(msleep$awake)

## [1] 5.9

# gets the interquartile range of awake variable  
mean(msleep$awake)

## [1] 13.56747

# calculating mean of awake variable  
median(msleep$awake)

## [1] 13.9

# calculating median of awake variable  
var(msleep$awake)

## [1] 19.82106

# calculating Variance of awake variable

# Displaying summary of dataframe msleep  
summary(msleep)

## name genus vore order   
## Length:83 Length:83 Length:83 Length:83   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
##   
##   
##   
##   
## conservation sleep\_total sleep\_rem sleep\_cycle   
## Length:83 Min. : 1.90 Min. :0.100 Min. :0.1167   
## Class :character 1st Qu.: 7.85 1st Qu.:0.900 1st Qu.:0.1833   
## Mode :character Median :10.10 Median :1.500 Median :0.3333   
## Mean :10.43 Mean :1.875 Mean :0.4396   
## 3rd Qu.:13.75 3rd Qu.:2.400 3rd Qu.:0.5792   
## Max. :19.90 Max. :6.600 Max. :1.5000   
## NA's :22 NA's :51   
## awake brainwt bodywt   
## Min. : 4.10 Min. :0.00014 Min. : 0.005   
## 1st Qu.:10.25 1st Qu.:0.00290 1st Qu.: 0.174   
## Median :13.90 Median :0.01240 Median : 1.670   
## Mean :13.57 Mean :0.28158 Mean : 166.136   
## 3rd Qu.:16.15 3rd Qu.:0.12550 3rd Qu.: 41.750   
## Max. :22.10 Max. :5.71200 Max. :6654.000   
## NA's :27

# Displays summary of one variable in a single row  
summary(msleep$sleep\_total)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.90 7.85 10.10 10.43 13.75 19.90

# from data set selecting two variable and summarizing  
msleep %>%  
 # selecting two variables from msleep and using pipe operator displaying summary  
 select(sleep\_total,brainwt) %>%   
 summary #getting the summary

## sleep\_total brainwt   
## Min. : 1.90 Min. :0.00014   
## 1st Qu.: 7.85 1st Qu.:0.00290   
## Median :10.10 Median :0.01240   
## Mean :10.43 Mean :0.28158   
## 3rd Qu.:13.75 3rd Qu.:0.12550   
## Max. :19.90 Max. :5.71200   
## NA's :27

# A summary table is created for vore values  
# showing the min, max, and the difference and arranging data set by the average  
msleep %>%  
 drop\_na(vore) %>%   
 # drops the missing values  
 group\_by(vore) %>%   
 # grouping by vore variable  
 summarise(  
 # finding the measures and assigning new reference for summarizing  
 Lower = min(sleep\_total),   
 # min of sleep\_total  
 Average = mean(sleep\_total),   
 #mean of sleep\_total  
 Upper = max(sleep\_total),  
 # max of sleep\_total  
 Difference = max(sleep\_total) - min(sleep\_total)  
 # difference in max and min of sleep total   
 ) %>%  
 arrange(Average)

## # A tibble: 4 × 5  
## vore Lower Average Upper Difference  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 herbi 1.9 9.51 16.6 14.7  
## 2 carni 2.7 10.4 19.4 16.7  
## 3 omni 8 10.9 18 10   
## 4 insecti 8.4 14.9 19.9 11.5

# arranges data set according to Average

# creating contingency tables  
library(MASS) # loading the mass package

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

attach(Cars93) # can apply base R functions on data set Cars93  
glimpse(Cars93) # displays quick overview of the dataset

## Rows: 93  
## Columns: 27  
## $ Manufacturer <fct> Acura, Acura, Audi, Audi, BMW, Buick, Buick, Buick,…  
## $ Model <fct> Integra, Legend, 90, 100, 535i, Century, LeSabre, R…  
## $ Type <fct> Small, Midsize, Compact, Midsize, Midsize, Midsize,…  
## $ Min.Price <dbl> 12.9, 29.2, 25.9, 30.8, 23.7, 14.2, 19.9, 22.6, 26.…  
## $ Price <dbl> 15.9, 33.9, 29.1, 37.7, 30.0, 15.7, 20.8, 23.7, 26.…  
## $ Max.Price <dbl> 18.8, 38.7, 32.3, 44.6, 36.2, 17.3, 21.7, 24.9, 26.…  
## $ MPG.city <int> 25, 18, 20, 19, 22, 22, 19, 16, 19, 16, 16, 25, 25,…  
## $ MPG.highway <int> 31, 25, 26, 26, 30, 31, 28, 25, 27, 25, 25, 36, 34,…  
## $ AirBags <fct> None, Driver & Passenger, Driver only, Driver & Pas…  
## $ DriveTrain <fct> Front, Front, Front, Front, Rear, Front, Front, Rea…  
## $ Cylinders <fct> 4, 6, 6, 6, 4, 4, 6, 6, 6, 8, 8, 4, 4, 6, 4, 6, 6, …  
## $ EngineSize <dbl> 1.8, 3.2, 2.8, 2.8, 3.5, 2.2, 3.8, 5.7, 3.8, 4.9, 4…  
## $ Horsepower <int> 140, 200, 172, 172, 208, 110, 170, 180, 170, 200, 2…  
## $ RPM <int> 6300, 5500, 5500, 5500, 5700, 5200, 4800, 4000, 480…  
## $ Rev.per.mile <int> 2890, 2335, 2280, 2535, 2545, 2565, 1570, 1320, 169…  
## $ Man.trans.avail <fct> Yes, Yes, Yes, Yes, Yes, No, No, No, No, No, No, Ye…  
## $ Fuel.tank.capacity <dbl> 13.2, 18.0, 16.9, 21.1, 21.1, 16.4, 18.0, 23.0, 18.…  
## $ Passengers <int> 5, 5, 5, 6, 4, 6, 6, 6, 5, 6, 5, 5, 5, 4, 6, 7, 8, …  
## $ Length <int> 177, 195, 180, 193, 186, 189, 200, 216, 198, 206, 2…  
## $ Wheelbase <int> 102, 115, 102, 106, 109, 105, 111, 116, 108, 114, 1…  
## $ Width <int> 68, 71, 67, 70, 69, 69, 74, 78, 73, 73, 74, 66, 68,…  
## $ Turn.circle <int> 37, 38, 37, 37, 39, 41, 42, 45, 41, 43, 44, 38, 39,…  
## $ Rear.seat.room <dbl> 26.5, 30.0, 28.0, 31.0, 27.0, 28.0, 30.5, 30.5, 26.…  
## $ Luggage.room <int> 11, 15, 14, 17, 13, 16, 17, 21, 14, 18, 14, 13, 14,…  
## $ Weight <int> 2705, 3560, 3375, 3405, 3640, 2880, 3470, 4105, 349…  
## $ Origin <fct> non-USA, non-USA, non-USA, non-USA, non-USA, USA, U…  
## $ Make <fct> Acura Integra, Acura Legend, Audi 90, Audi 100, BMW…

# table () for the variable Origin  
table(Origin)

## Origin  
## USA non-USA   
## 48 45

# summary of origin of cars displays  
  
# table() for multiple variables  
table(AirBags, Origin)

## Origin  
## AirBags USA non-USA  
## Driver & Passenger 9 7  
## Driver only 23 20  
## None 16 18

# keeping the above table function to the addmargins()  
addmargins(table(AirBags,Origin),1)

## Origin  
## AirBags USA non-USA  
## Driver & Passenger 9 7  
## Driver only 23 20  
## None 16 18  
## Sum 48 45

# 1 in the end of the function sums the rows  
addmargins(table(AirBags,Origin),2)

## Origin  
## AirBags USA non-USA Sum  
## Driver & Passenger 9 7 16  
## Driver only 23 20 43  
## None 16 18 34

#If there is 2 in the end it sums up the columns

table(AirBags, Origin)

## Origin  
## AirBags USA non-USA  
## Driver & Passenger 9 7  
## Driver only 23 20  
## None 16 18

# generates contingency table of AirBags and Origin  
prop.table(table(AirBags, Origin))

## Origin  
## AirBags USA non-USA  
## Driver & Passenger 0.09677419 0.07526882  
## Driver only 0.24731183 0.21505376  
## None 0.17204301 0.19354839

# Calculate the proportion of AirBags and Origin  
prop.table(table(AirBags, Origin))\*100

## Origin  
## AirBags USA non-USA  
## Driver & Passenger 9.677419 7.526882  
## Driver only 24.731183 21.505376  
## None 17.204301 19.354839

# converting proportions to percentages by multiplying previous with 100  
round(prop.table(table(AirBags, Origin), 2)\*100)

## Origin  
## AirBags USA non-USA  
## Driver & Passenger 19 16  
## Driver only 48 44  
## None 33 40

# round the percentage proportions to two decimal places for precision

#it gets messy to avoid that Greg suggested to use tidyverse with the pipe operators  
#Below is the example for same  
Cars93 %>%  
 group\_by(Origin, AirBags) %>%   
 # groups by Origin and AirBags  
 summarise(number = n()) %>%   
 # Summarizes the data by count of number of occurrences  
 #pivot the data wider function selects the names from a origin variable  
 #and values from number variable  
 pivot\_wider(names\_from = Origin,  
 values\_from = number)

## `summarise()` has grouped output by 'Origin'. You can override using the  
## `.groups` argument.

## # A tibble: 3 × 3  
## AirBags USA `non-USA`  
## <fct> <int> <int>  
## 1 Driver & Passenger 9 7  
## 2 Driver only 23 20  
## 3 None 16 18